

AMENDMENTS TO THE CLAIMS

Please amend the Claims as follows:

1. (Original) A computer implemented method of creating and managing one or more interceptors, comprising:

intrinsically chaining said interceptors, said interceptor chains made up of dissimilar types of interceptors,

storing state information, in at least one of the chained interceptors, directed to a reference to a next interceptor, and

providing a unified binding mechanism across request level Object Request Broker (ORB) services, message level ORB services, and transports.
2. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 1, wherein said chained interceptors are contiguous or split.
3. (Currently Amended) A computer implemented method of creating and managing one or more interceptors, as per claim 2, further comprising:

splitting, in a ~~server~~server, the chained interceptors into first and second
interceptor

chains.

4. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 3, wherein said first interceptor chain is a per-client interceptor chain.
5. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 3, wherein the second interceptor chain is one of a per-endpoint and a per-object interceptor chain.
6. (Original) A computer implemented method of creating and managing one or more interceptors, comprising:
intrinsically chaining said interceptors,
storing state information, in at least one of the chained interceptors, directed to a reference to a next interceptor,
providing callbacks within said interceptor chain, and
whereby a request only has to traverse said chain once while allowing interceptors that need to intercept processing at additional points to do so by wrapping a callback interface.
7. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 6, wherein wrapper server request callback implementations passed along by intermediate server request interceptors perform their own processing as well as delegate operations back to encapsulated server request callback instances.

8. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 6, wherein an ORB service's implementation of a server request callback interface allows it to intercept, observe, or influence outcomes of the requests before delegating back to the server request callback instance that it wraps.
9. (Cancelled).
10. (Currently Amended) A computer implemented method of creating and managing one or more interceptors, as per claim 6, wherein server request callback implementations provided by ~~orb~~-ORB services delegate read inputs operations back to the server request callback instances they wrap, unless they are causing an exception to be returned.
11. (Currently Amended) A computer implemented method of creating and managing one or more interceptors, as per claim 6, wherein server request callback implementations provided by ~~orb~~-ORB services delegate write output operations back to the server request callback instances they wrap.
12. (Original) A computer implemented method of creating and managing one or more interceptors, comprising:

intrinsically chaining said interceptors into one or more chains,
storing state information, in at least one of the chained interceptors, directed to a
reference to a next interceptor,
applying a flyweight pattern to an Interoperable Object Reference (IOR)
representation, said flyweight pattern implementing policies which control which
chain of interceptors to use at invocation.

13. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein said IOR representation comprises any of: IORs, IOR profiles, and IOR components, and a binding mechanism hashes and compares IORs, IOR profiles, and IOR components using their pointers instead of their content.
14. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein said flyweight pattern is applied to multiple IOR interfaces.
15. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein for each implementation of said IOR interfaces, no more than a single instance with a particular value is instantiated at a time.

16. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein when an IOR is demarshaled multiple times, a single instance is shared.
17. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein when an item being demarshaled is instantiated, no heap allocation is required.
18. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein when said flyweight pattern is applied, a map of values currently instantiated is maintained.
19. (Original) A computer implemented method of creating and managing one or more interceptors, as per claim 12, wherein when said flyweight pattern is applied, scalability is insured by insuring that redundant copies of any of: individual components, profiles, and IORs are not stored in memory.